

Enigma recoding file format

This document describes the Enigma recording file format. Enigma recording refers to continuous recording of flight data to a recording file on an SD/MMC card inserted into an Enigma compatible instrument (EFIS).

This file format has been created to support the Enigma series of EFIS instruments from MGL Avionics.

MGL Avionics grants any interested party the right to use this file format whether this is in support of a MGL Avionics product or not.

Any party making use of this document and described file format will do so at their own discretion, responsibility and risk.

The party further acknowledges that any data it receives howsoever in this format may be faulty, incorrect, incomplete or outdated and that the party bears responsibility for any consequence that may arise out of this.

MGL Avionics places this document and the data format it describes in the public domain.

In order to allow recognition of the format any party adopting this format agrees to refer to this format as the “Enigma recording format”. All data using this format should be considered to be in the public domain and the implementer will make reasonable effort to allow any interested party to use the data free of cost with the exception of a reasonable fee for distribution and media.

Exceptions:

MGL Avionics reserves copyright and denies use of this data format for any military or related purpose.

Comments are welcome: info@MGLAvionics.co.za

Note:

This file format is valid from version 0.1.2.6 of Enigma application software.

The recording file

Recording takes place onto a pre-created, fixed length recording file. The recording file initially starts as a file filled with zero bytes. The file size, recording frequency and recording content determines the length of recording time.

Recording is “endless”. When the end of the file is reached, recording proceeds at the start of the file, overwriting oldest recorded data. Data records are never fragmented. If the recording reaches the end of the file and there is not enough space left to write the current record, the recording proceeds at the beginning of the file.

Using a fixed length, pre-created file avoids the need for the operating system to write to directory and file allocation structures. This prevents inadvertent loss or corruption of the file system in a live system. It is expected that the SD/MMC card can be removed and inserted at any time.

Records

Currently, the following records are defined:

- a) Primary flight data
- b) Attitude
- c) Engine data related to RDAC 1
- d) Engine data related to RDAC 2
- e) GPS data

Primary Flight data

The primary flight data record has the following format:

Altitude	longint	feet, can go negative
Barometer	smallint	millibars – ambient pressure
Airspeed	smallint	statute miles/hour
TAS	smallint	true airspeed in statute miles/hour
VSI	smallint	vertical speed in feet/minute. Positive and negative
Glide	smallint	glide ratio in tenths. Positive and negative
RotorRPM	word	RPM from rotor input (bit 15 is digital level at rotor input)
MainVoltage	byte	voltage in tenth of a volt
BackupVoltage	byte	voltage in tenth of a volt
Current	smallint	current in tenth of an amp. Positive and negative
AOA	smallint	Angle of attack

AmbientTemp smallint degrees C. Positive and negative

.

Attitude data

The attitude data record has the following format

BankAngle:	smallint	+/-180 degrees
PitchAngle	smallint	+/-90 degrees
Slip	smallint	+/-50
CompassHeading	smallint	magnetic heading
Gyro_Heading	word	yaw 0-359 degrees
GForce	smallint;	in tenth of a G, positive and negative
TurnRate	smallint;	degrees/minute, positive and negative

GPS data

The GPS data record has the following format:

GPS_Latitude	single	decimal format IEEE float
GPS_Longitude	single	decimal format IEEE float
GPS_HeadingValue	longint	track in whole degrees 0-359
GPS_GroundSpeed	longint	statute miles per hour
GPS_Altitudevalue	longint	feet (not valid unless in 3D mode)
GPS_Status	byte	0 = GPS acquiring 1 = Dead reckoning mode 2 = 2D fix 3 = 3D fix
GPS_Satellites	byte	Number of satellites tracked
GPS_HAcc	byte	Horizontal accuracy estimate in feet (radius)
GPS_VAcc	byte	Vertical accuracy estimate in feet

Note: Software version prior to 0.1.2.6 does not have the last four bytes.

Engine data

The engine data record has the following format (identical RDAC1 and RDAC2)

RPM	word	
TankReading1	word	raw tank reading 0-4095
TankReading2	word	raw tank reading 0-4095
CHTOne	word	degrees C (Rotax 912 CHT)

CHTTwo	word	degrees C (Rotax 912 CHT)
FuelFlow	word	tenth of a liter / hour
MAP	word	millibars
FuelLevel1	word	whole liters
FuelLevel2	word	whole liters
FuelLevelCalculated	word	tenth of a liter
OilTemp	word	degrees C
OilPressure	word	tenth of a bar
CarbWarn	smallint	degrees C, can go negative
FuelPressure	byte	tenth of a bar
WaterTemp	byte	degrees C
EGT1-EGT12:	word (12 values)	degrees C
RDACTemp	word	raw RDAC temperature used for cold junction compensation
RDACFail	boolean	true if no data from RDAC

Recording packet structure

A recording packet is a compound structure consisting of some or all of the available records. Records may be individually enabled or disabled for recording by the EFIS. Typically, at least the primary flight data will be recorded (This cannot be disabled).

A recording packet starts with the following two header bytes:

0xAA

0x55

This is followed by a byte containing the total number of bytes in the packet (excluding the two header bytes)

Length of packet data

This is followed by a byte containing the length of the primary flight data record plus length of the time stamp

Length of primary flight data record plus time stamp

This is followed by a time stamp of four bytes length (longint). The time stamp contains the number of seconds elapsed since the 00:00 1st january 2000.

Time stamp (4 bytes)

Primary flight data packet (24 bytes)

If RDAC1 recording is enabled, this is followed by:

byte 0x01

byte length of engine data packet

RDAC 1 data packet (55 bytes)

If RDAC2 recording is enabled, this is followed by:

byte 0x02

byte length of engine data packet

RDAC 2 data packet (55 bytes)

If attitude recording is enabled, this is followed by:

byte 0x03

byte length of attitude data packet

attitude data packet (14 bytes)

If GPS recording is enabled, this is followed by:

byte 0x04

byte length of GPS data packet

GPS data packet (20 bytes)

If insufficient space remains at the end of the file to write the current packet, an end-of-recording marker is written if at least two bytes remain. This takes the value of two bytes = 0xBB 0xDD. Recording of the current record begins at the start of the file.

Synchronizing to an existing recording file

In order to find the last record in a file (most currently recorded) proceed as follows:

Start reading 8 bytes from the start of the file. If a recording exists, you will read the start of packet header (0xAA 0x55), size of packet and time stamp.

Using this information, seek to the next packet and read the header. If you can read a valid header, compare the time stamps. If the time stamp of the second packet predates the one from the first, you have found the end of the recording. If you cannot read a valid header, you have found the end of recording.

Continue until you find the “oldest” end of recording or end of file. The next recording packet would start at the current location, overwriting an old packet or creating a new packet.

A similar method is used to find the oldest packet, for example as a start to a data playback. If you cannot find an old packet preceded by a newer one, the oldest packet is the first packet in the file.

Exceptions

packets are tagged by a time stamp. This time stamp is obtained from the real time clock of the EFIS. It is possible that time is adjusted from time to time and it may be adjusted backwards. This causes a temporary problem with the recording as we are creating a situation where the time stamp “goes backwards” from one recording to the next. This can be interpreted as having found the end of recording.

As such an adjustment would typically be a few minutes at most, this should be taken into account during scanning to avoid false detection.

Should time be adjusted backwards by a large amount, it would normally be required to start a new recording file.